SSRPT (SSR POINTER TRACKER) FOR CASSINI MISSION OPERATIONS - A GROUND DATA ANALYSIS TOOL

Edwin P. Kan

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, Ca. 91109, USA

(edwin.p.kan@jpl.nasa.gov)

ABSTRACT

Tracking the resources of the two redundant Solid State Recorders (SSR) is a necessary routine for Cassini spacecraft mission operations. Instead of relying on a full-fledged spacecraft hardware / software simulator to track and predict the SSR recording and playback pointer positions, a stand-alone SSR Pointer Tracker (SSRPT) tool was developed as part of JPL's Multimission Spacecraft Analysis System. The SSRPT design, operational implementation, and customization for Cassini are described in this paper.

Keywords: Solid State Recorder; Pointer position tracking; Position prediction; Cassini spacecraft; Ground analysis software

1. INTRODUCTION

The Solid State Recorder Pointer Tracker (SSRPT) is a ground data analysis "tool", developed for mission operations and analysis of the Solid State Recorders (SSRs) of the Cassini spacecraft.

Tracking the record_pointer and playback_pointer positions of the redundant SSR_A and SSR_B is a necessary routine in ground systems and mission operations. This is particularly true for the Cassini mission, because telemetry downlink time is very limited over the mission lifetime, and is of the order of 8 hours per 7 day interval. Telemetry downlink provides a mixture of real-time telemetry and pre-recorded telemetry, the latter being collected and recorded by the SSRs off-line.

Mission operations analysts are required to be able to pinpoint the address ranges of desired recorded telemetry for a specific recording time period corresponding to certain spacecraft activities. Otherwise, the precious downlink bandwidth and infrequent schedule may be wasted.

On-board command sequences and uplinked real-time commands can affect the operation of the SSR's. Their effects on SSR pointer positions need to be predicted. Mission analysts need to prevent overwriting on data that has not been playbacked, and need to promote the most efficient use of SSR memory.

For the above purposes, a stand-alone ground analysis tool, the SSRPT, was developed. This is an inexpensive substitute of a full-fledged hardware/software simulator of the spacecraft. SSRPT was developed as part of a the Multimission Spacecraft Analysis System (MSAS), under the auspices of the Jet Propulsion Laboratory Multimission Ground Systems Office (Ref. 1, 2).

2. DESIGN CONCEPTS AND DEVELOPMENT NOTES

The Development Process

Software development of SSRPT follows the classic Yourdon's software development methodology. High level user requirements (Ref. 3) provided the starting point of the process. Detailed functional requirements and software requirements were then developed, in a more compact form of a "Package Requirements Document" (Ref. 4). The responsible party of this latter product was called the "User Representative", who refined high level user requriements (and sometimes desirements), distilled the engineering facts, and converted them into functional designs.

There was a less-than-totally formal software design process before the C coding. This reduced-effort process was compensated by good data flow diagram, comprehensive pseudo code, and detailed data dictionaries generated by the User Representative. Test cases and test procedures were also developed by the User Representative. Test cases were generated with the aid of several spreadsheet tools, which do not incorporate the in-depth logic embodied in SSRPT. In the software coding process, standard X-window GUI (Graphics User Interface) widgets and common API's (Application Interfaces) were used - SSRPT being one of the dozens of tools developed by MSAS.

It is under this spirit that dozens of other tool are developed in JPL's MSAS (Ref. 1), including the Flight Software Memory Tracker (Ref. 5).

SSRPT General Design Description

Figure 1 is the SSRPT context diagram. In this diagram, the interaction between SSRPT and spacecraft sequences is evident. However, in decomposing the functionality of SSRPT, it becomes necessary to construct the basic "Calculator" program, which forms the core program to be called by a program that parses spacecraft commands. Hence, SSRPT is designed as two separate tools: (a) ssrptcalc, the "Calculator", and (b) ssrptsp, the "Sequence Predictor".

ssrptcalc behaves like a calculator, implemented in a GUI (Graphic User Interface), which provides the basic computation of delta_time multiplied by recording and playback bit_rates (rec_rate and PB_rate). Cassini has multiple telemetry modes and submodes, each having its own record and playback rates. The SSR has four operation modes, of which the FIFO (First_In_First_Out) mode is the default operation mode.

ssrptcalc is actually more than a calculator. ssrptcalc models certain logics design features of the SSR. The most prominent feature, particularly true for the FIFO mode, is the inhibition of recording new data which would overwrite data addresses that have not been playbacked. This inhibition is effected via a "priority" state, implemented by the so-called lap_bits.

ssrptsp is a sequence predictor by the fact that it predicts a profile of the SSR pointer positions according to the timeline of commands in a spacecraft sequence. When a command does not cause a change in SSR states and pointer repositioning, ssrptsp simply propagates pointer positions in the normal fashion. When a command does cause a change in SSR states and/or pointer positions, ssrptsp parses the command, causes the state changes accordingly, calls the ssrptcalc and other programs as required, before further pointer propagation.

ssrptcalc is a stand-alone program, written in C language, and wrapped around X-windows and "motif" widgets (for the GUI presentation). ssrptsp is designed around the SEQGEN program set, developed and long used by the mission planners and operations analysts at the Jet Propulsion Laboratory (discussed in numerous journal and conference papers) (Ref. 6). Specific ssrptsp logics are incorporated in the form of "spacecraft model files" (smf) and "context variable files" (cvf).

3. ssrptcalc - The SSRPT Calculator

Figure 2 is the Level 1 ssrptcalc Data Flow Diagram. Since this is more than a subroutine to be called by ssrptsp, and since it is to be a standalone calculator, Bubble 1.1 embodies the User Input Setup Configuration. In this bubble are resident all the input parameters as shown in the GUI representation of the calculator, Figure 3.

The calculator function is basically:

new_position = old_position + rate x elapsed_time

Bubble 1.2 provides the rate, which comprises the record_pointer rate and the playback_pointer rate. These rates depend on the "Telemetry Mode"; the selection of the mode is via the GUI, and the specification of the corresponding rate is in a TLMSSRR (telemetry rate table) lookup table. Other default parameters are specified in "cvf" (Context Variable File) files.

Bubble 1.3 computes the elapse_time, over which the position change is to be evaluated. Initial time and final time or delta time are input parameters, as accepted on the GUI. Both scet (spacecraft event time in UTC time) and sclk (spacecraft clock) time can be input in the time field. Delta time is input in hh:mm:ss.sss format. Conversion between scet and sclk time is performed here, when necessary.

Bubble 1.4 is the position change computation. Here reside the logic of SSR end-around wrap around, the carry-over of "lap_bits", the detemination of record pointer taking over the playback pointer (or vice versa), the preservation of "record priority over playback priority" (or vice versa, depending on the case), the truncation of pointer positions according to "frame" boundaries, and production of miscellaneous diagnostics and warning (error) messages.

Bubble 1.5 embodies the reporting and presentation of output parameters, including the record pointer position, playback pointer position, final configuration of "lap_bits", and output messages on the GUI and on a text output report. The latter report is particularly useful as a development diagnostics tool.

The sample calculation in Figure 3 shows the propagation of record pointer and playback pointer over the duration of 6 hrs 12 min 59.999 sec, where the Telemetry Mode is SAF_142200 (a checkout mode at 142200 bps), at submode "REU_norm", corresponding to a record_rate of 143.2 kbps, and playback_rate of 121.8 kbps. The SSR is in the FIFO mode, and has the partition size of 125,829,112 words. The final positions and lap_bits are shown in the lower portion of the calculator. The change in the initial lap_bits (11) to the final lap_bits (00) indicates the rollover of both record and playback pointers. An external message annuciates this rollover of both pointers. Printing the output file (a text formatted file) is achieved by clicking the print-icon on the GUI, and submitting the proper file redirection in subsequent motif windows.

4. ssrptsp - the Sequence Predictor

Figure 4 is the Level 1 ssrptsp Data Flow Diagram. It is very similar to the ssrptcalc Data Flow Diagram of Figure 2, with the exception that the process is now repeated for each and everyone of the commands in a spacecraft sequence. For those commands that are parsed and recognized to have no effects on SSR configuration or on SSR record and playback pointer positions, no call is made to the ssrptcalc. For those commands that are parsed and recognized to cause SSR configuration and pointer changes, the affected variables are updated; special logic is used where necessary. Elapse_time is computed as that time between two commands that cause positive effects on the SSR, and over which time the record_pointer and playback_pointer positions are propagated via a call to the ssrptcalc.

Figure 5 shows the ssrptsp GUI representation. This GUI holds the entries of various input files, including smf, sasf, cvf and environment files,. On the toolbar are icons which invoke a text file editor, Environmental File Editor, the SEQGEN State Tracker engine, the ssrptcalc, among other typical file and resource handling utilities.

The parsing of commands is actually accomplished via a well established program set, SEQGEN, used by JPL mission sequence analysts. The adaptation of SEQGEN to the present ssrptsp requirements is achieved via the development of special "smf" (spacecraft model file) and cvf files.

The ssrpt.smf file is adapted for ssrptsp, and contains the definitions of all spacecraft commands as found in the "Command Dictionary". The response of the spacecraft to each and everyone of these commands is modeled, causing state changes and invocation of ssrptcalc and other program calls. Error messages, where appropriate, are also generated. Initial condition files and user inputs via cvf files are modeled and invoked by SEQGEN.

The following exposition excerpts some relevant details from smf, cvf, sasf and pef files, in order to illustrate the process and design of ssrptsp:

****

Sample lines from ssrpt.smf are as follows:

```
smf File Header
$$EOH
SSRPT(telemetry,
  ATTRIBUTES,
    sc_mode(TYPE,STRING,
       RANGE,\"RTE_5","RTE_10","RTE_20",...
       UNKNOWN","TEST1","TEST2","TEST3"\,
       SHOW,YES),
    cds_mode(TYPE,STRING,
       RANGE,\"REU_NORM","V7",...,"TEST"\,
       SHOW,YES),
    ...
       end,
  COMMANDS,
    6CHG_SC_TM_DIR(TITLE, PARAMETERS ...
       RESULTS,
         CALL,SSRevent(),
         IF,SSRPT_control(...
           CALL,tlmReset(), ...
         end,
         ELSE, E$ERROR="SSRPT is not ...",
           CALL,setSSRPTabort("SEQ"),
         end,
       end),
  SUBROUTINES,
    rates(...)
SSRPT(SSR_A,
    ATTRIBUTES ..
    COMMANDS ...
    SUBROUTINES ... )
  ...
SSRPT (control ... )
SSRPT(calculator_interface,
    SUBROUTINES,
       SSRevent ...
       calculator ...
SSRPT(_InitialCondModel ... )
$$EOF
```

Sample lines from ssr_constants.cvf file are as follows:

```
cvf File Header
$$EOH
/SSR_constants
"bits_per_word"        16
"words_per_frame"      550
"min_partition_size"   3
"max_partition_size"   134217727
$$EOF
```

Exemplifying with sample lines from in.sasf (spacecraft activity sequence file):

```
sasf File Header
$$EOH
request(SSR_activity,
    START_TIME, 1998-002T00:00:00.000,
    REQUESTOR, "barbara",
    DESCRIPTION, "TLM Mode Check",
    PROCESSOR, "PRI",
    KEY, "SUB06")
  command(1,
    SCHEDULED_TIME,\0:0:0\,FROM_PREV_START,
    6CHG_SC_TM_DIR(0,"LOCAL")),
  command(2,
    SCHEDULED_TIME,\2:0:0\,FROM_PREV_START,
    6CHG_CDS_TM_DIR(0,"LOCAL")),
    ...
  command(6,
    SCHEDULED_TIME,\2:0:0\,FROM_PREV_START,
    6SSR_MLD_COPY("SSR_A","PART_0","SSR_B",
       "PART_0")),
end;
request(...)
...
$$EOF
```

The ssrptsp outputs the following responses in the out.pef (predict event file):

```
pef File Header
$$EOH
1262296821:000 1997-365T22:00:00.000 CDS:
  prime=A,connected_SSR=A,unconnectd_SSR=B,
  SSR_A_toCDS=A,SSR_B_toCDS=B;
...
1262296821:000 1997-365T22:00:00.000
  SSR_A_PART_4: created=TRUE,deleted=FALSE,
  write_protected=FALSE,partition_size=
  125829000.0, partition_mode=3,recording=
  TRUE,playing_back=TRUE,NSTAT15_rec_
  lapbits=00,NSTAT15_PB_lapbits=00,base_
  ptr_time=1997-365T22:00:00.000,base_ptr_
  event=SEQUENCE _start,base_rec_ptr_pos_
  word=0.0,base_PB_ptr_pos_word=0.0,init_
  rec_ptr_time=1997-365T22:00:00.000,init
  _rec _ptr_pos_word=0.0,rec_ptr_pos_bit=
  0.0,rec_ptr_pos_ word=0.0,rec_ptr_pos_
  TF=0,init_PB_ptr_time=1997-365T22:00:00.
  000,init_PB_ptr_ pos_word=0.0,PB_ptr_
  pos_bit=0.0,PB_ptr_pos_word=0.0,PB_ptr_
  pos_TF=0;
...
1262419221:000 1998-002T08:00:00.000 CMD,
  6CHG_SC_TM_IMM,PRI,SSR2$3CMD,RTE_40; <<
  IMMED S/C TLM MODE CHG >>;
1262419221:000 1998-002T08:00:00.000
  SSR_A_PART_4:..,rec_ptr_pos_TF=112514600,
  PB_ptr_pos_TF=217800 ...
1262419221:000 1998-002T08:00:00.000
  telemetry: sc_mode=RTE_40,cds_mode=REU_
  NORM,rate_table_available=TRUE,rec_
  rate=1635.86999511,PB_rate=17.41399955;
...
$$EOF
```

Using another in-house tool (or other editing tool, such as written via PERL scripts), the following is a tabular form of the predicted playback and record pointer positions ordered by scet time:

scet	rec_ptr	PB_ptr
1997-365T22:00:00.000	0	0
1997-365T23:59:57.336	100735800	92400
1998-001T00:59:36.670	101101550	96250
1998-001T01:59:16.003	101467300	100100
1998-001T02:58:55.336	101833600	104500
1998-001T03:58:34.670	102199350	108350
...		
1998-002T08:00:03.336	112514600	217800
...		

****

It is straightfoward to convert the tabular output into graphic presentations using off-the-shelf or custom plotting routines.

The above exposition illustrates how an analyst can take a spacecraft command file, feed it through the ssrptsp (by invoking SEQGEN), use initial conditions file (not illustrated here), have ssrptsp output a predicted event file (the output), a final conditions file (not illustrated here), have the output file post-processed to obtain tabular or graphic plots of the SSR record_pointer and playback_pointer positions.

5. SUMMARY

SSRPT has been used routinely for data analysis and sequence analysis by Cassini CDS (Command and Data Subsystem, the subsystem that holds cognizance over the SSRs) mission operations analysts during pre-launch phases (through October 15, 1997). It is being used routinely after launch, and is expected to be used throughout the lifetime of Cassini. For the expressed purpose, this standalone tool obviates the use of full-fledged hardware/software simulators. It is cost effective, is user friendly, and is operational. It design architecture permits engineering changes via modification of lookup tables, smf model files and cvf parameter files; and lends itself to ready adaptation to other spacecraft missions.

Acknowledgement

This work was carried out at the Jet Propulsion Laboratory (JPL), California Institute of Technology, under contract to the National Aeronautics and Space Administration. The functional requirements of this work were developed by R. Morillo; engineering design / software specification by E. Kan; software implementation by S. Petrosian and B. Larson.

REFERENCES

1. Wilson, Robert. K., and M. Hill, M., "Spacecraft Analysis, MSAS (Multimission Spacecraft Analysis System) - A Multi-Mission Solution," Paper #5f001, Proc. SPACEOPS 1998, 5th Int. Symp. on Space Mission Operations and Ground Data Systems, Tokyo, Japan, Jun. 1-5, 1998; also Document #JPL D-9173, Rev. D, (Functional Requirements Document), Jet Propulsion Laboratory, July 10, 1996.
2. Murphy, S.C., et.al., "Customizing the JPL Multimission Ground Data System," Proc. SPACEOPS 1994, 3rd Int. Symp. on Space Mission Operations and Ground Data Systems, held at GSFC, Greenbelt, Md., USA, Nov. 14-18, 1994.
3. Tapia, E. (custodian), "Cassini Functional Requirements 3-291, Uplink Formats & Command Tables," Jet Propulsion Laboratory Document #CAS-3-291, Rev. E, Jan. 24, 1997.
4. Kan, E. P., and H. Uffelman, "CDS (Command and Data Handling Subsystem) Package Requirements Document - Flight Software Memory Tracker," Jet Propulsion Laboratory Document #JPL D-9173 (Section 3.2), July 10, 1997.
5. Kan, E. P., and H. Uffelman, "Tracking Flight Software in Cassini Mission Operations Using the FMT Tool," Paper #5f005, Proc. of SpaceOps 1998, 5th Int. Symp. on Space Mission Operations and Ground Data Systems, Tokyo, Japan, Jun. 1-5, 1998.
6. Salcedo, J., "Advanced Multimission Operations System - Sequence Subsystem (SEQ) Version_19 SEQ_GEN User Guide," Jet Propulsion Laboratory Document #JPL D-111261, also JPL MOSO Document #W4_MSEQ0716-01-00-09, Dec. 1, 1993.

Figure 1. SSRPT Context Diagram (Level 0)

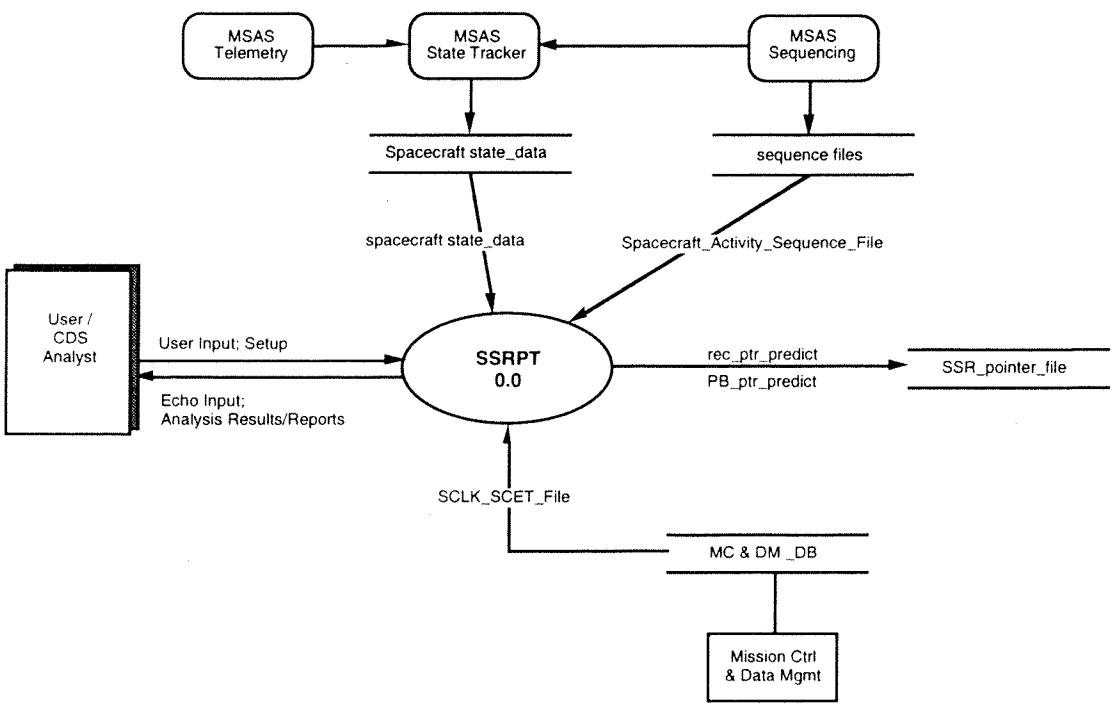


Figure 2. SSRPT Calculator (ssrptcalc) Data Flow Diagram (Level 1)

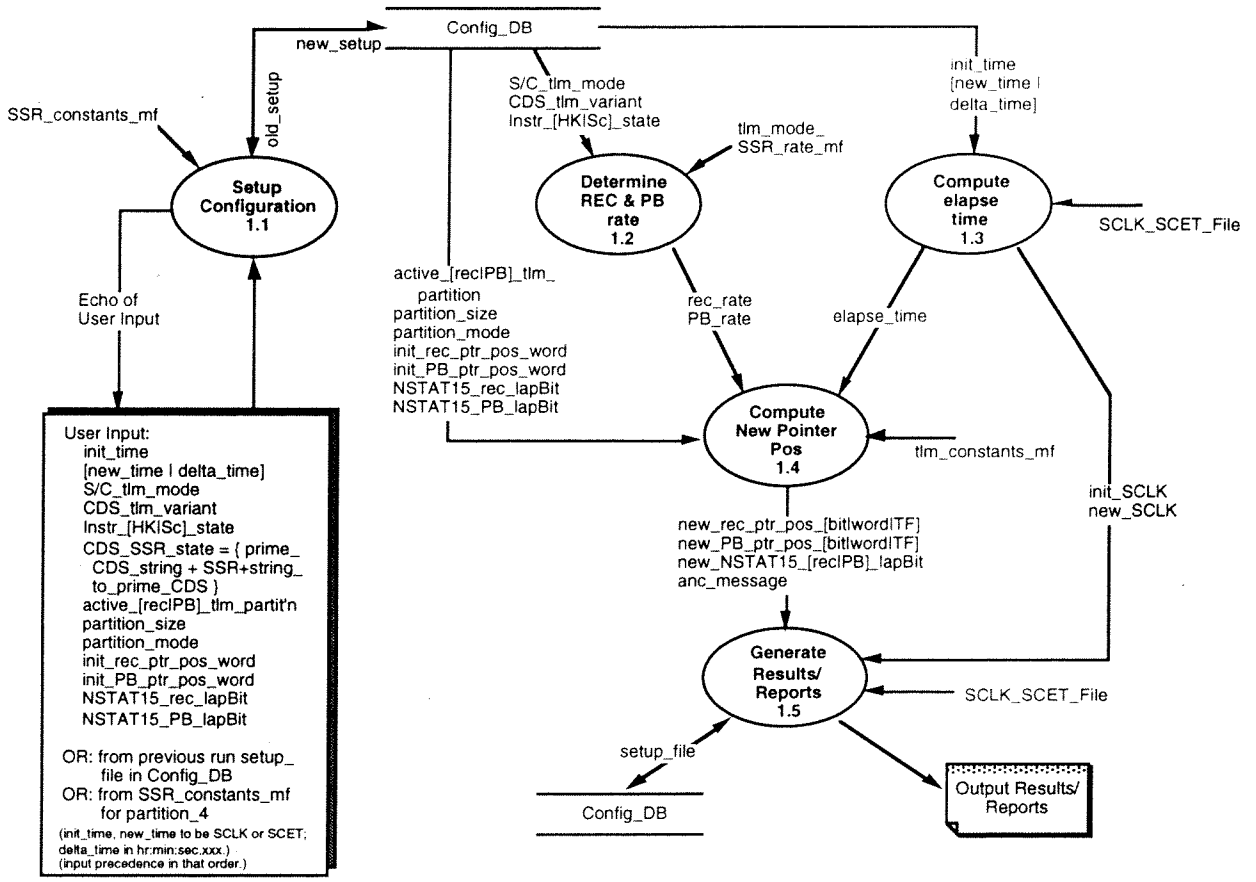


Figure 3. ssrptcalc GUI

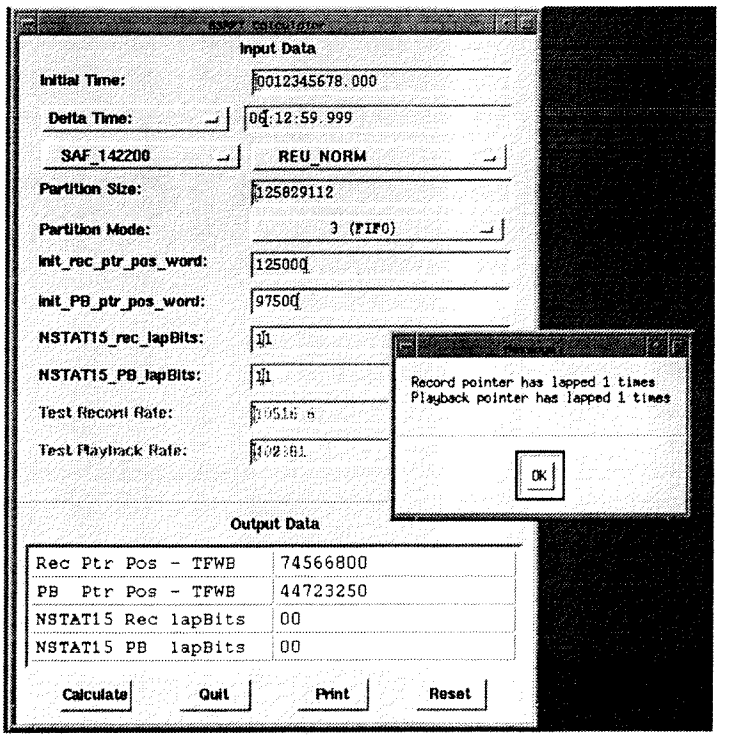
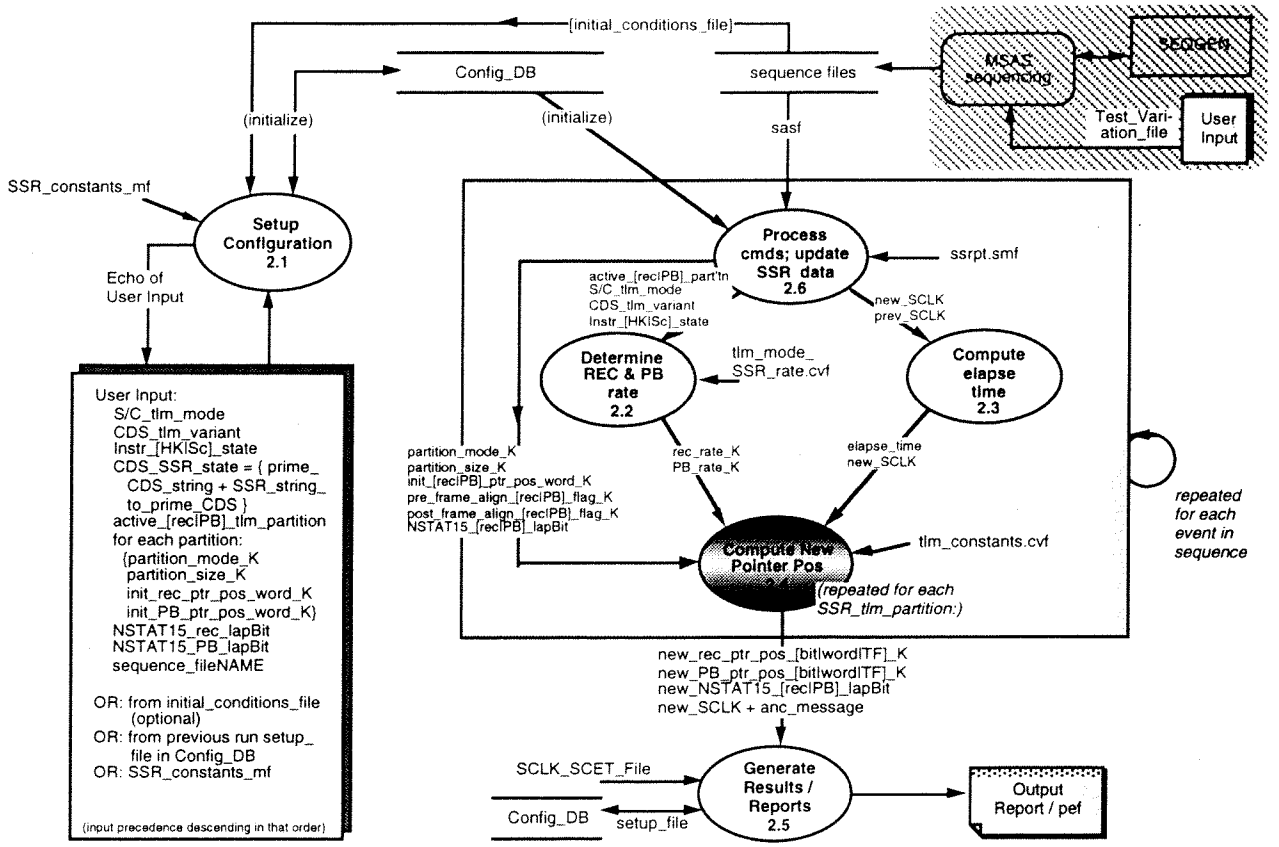


Figure 4. SSRPT Sequence Predictor (ssrptsp) Data Flow Diagram (Level 1)



This DFD assumes that SEQGEN, hence S/C State Tracker has the logic to predict major SSR states. SSRPT processes only a few 6_cmds.

Figure 5. ssrptsp GUI (with Environmental Editor Invoked)